Kevin Weng

Prof. Justin Tojeira

CSCI 335

4/10/24

	1st runtime	2nd run time	3rd run time
Vector	297 ms	360 ms	359 ms
List	2027 ms	876 ms	1996 ms
Heap	225 ms	224 ms	550 ms
AVL Tree	423 ms	747 ms	966 ms

Vector:

The vector file is designed to calculate the median of sequence of integers provided by the instructions vector. The vectorMedian function iterates through each element in the instructions vectors and it inserts or deletes based on the current value. When it comes across a -1, it will calculate and extract the median from the vector. The loop iteration iterates through each element in the instructions vector so it's time complexity is O(n). Insertions and deletions have an average time complexity of O(n). Finding the median takes O(1) because accessing a element by index in a vector can be done in constant time. Overall, the time complexity of myVector is O(n^2) because iterating through the instructions is O(n) and insertion/deletion is O(n) so it's going to be O(n^2).

List:

The time complexity of the iteration iterating through the instructions vector is O(n). When inserting elements, the time complexity is O(n) while when deleting, the time complexity is O(1) because you can just delete the element right away without looping or anything. The median

calculation time complexity is O(n) because unlike vectors, you need to traverse the list from the beginning until you reach the middle element. Overall the time complexity of myList is O(n^2) because for each element in instructions which is O(n) and with insertions or median calculations, they're both O(n) so overall it will be O(n^2).

Heaps:

The time complexity of the iteration iterating through the instructions vector is O(n). Both insertion and deletion's time complexity is O(log n). In heaps, we needed two different heaps, one to store values that is lower than the median and the other to store values that is greater than the median. The median calculation is O(1) per iteration. Overall the time complexity of myHeap is O(n * log n).

AVL tree:

The time complexity of the iteration iteration through the instructions vector is O(n). The run time insertions and deletions is O(log n) because we are looking at trees so it'll be faster to see where you need to insert or delete by seeing if the value is greater than less and to see if it should be inserted or deleted on left or right. The time complexity for finding the maximum and minimum is O(log n). Overall the time complexity of myAVLtree is O(n * log n).